

REMARKS

Applicant is in receipt of the Office Action mailed March 9, 2007. Claims 1, 3-7, 9-10, 12-24, 26-28, 30-32, 34-36, and 38-39 are pending in the case. Reconsideration of the present case is earnestly requested in light of the following remarks.

Objections

Claims 1, 23, 28, 32, and 36 were objected to for the phrase “the first portion of the program is a successively larger portion of the program”. The Examiner asserts that this is improper language usage in light of the Specification, citing p.8, middle paragraph, and p.11, middle paragraph. Applicant respectfully notes that the phrase actually reads “wherein for one or more iterations the first portion of the program is a successively larger portion of the program”, and that in the context of the independent claims, this simply means that during the iterative repetition, for some of the iterations, the portion of the program being deployed onto the programmable hardware element is a successively larger part of the program, i.e., more and more of the program is deployed to the programmable hardware element, as opposed to being executed on the computer system.

The middle paragraphs of p.8 describe this aspect of the claimed invention, as well as the case represented in claims 3, 26, 30, 34, and 38, specifically, “wherein for one or more other iterations the first portion of the program is a successively smaller portion of the program”:

In one embodiment, the steps described above may be repeated one or more times, where in each iteration the first portion of the program is a successively larger portion of the program. In other words, **the debugging process described above may be performed iteratively, where at each iteration the portion of the program being debugged (the remaining portion of the program) is a smaller portion of the program.**

It should be noted, however, that in other embodiments, one or more portions of the program that have been previously debugged may require further debugging, e.g., due to changes made in other parts of the program, and so may be re-incorporated into the remaining portion of the program. Thus, in one embodiment, the steps above may be repeated a plurality of times, where, **in a first subset of iterations the first portion of the program is a successively larger portion of the program, and in a second subset of iterations the first portion of the program is a successively smaller portion of the program. For example, the user may move some of the (previously debugged) first portion**

of the program back into the remaining portion for further debugging, thereby increasing the size of the remaining portion, while decreasing the size of the first portion. Of course, over the course of the debugging process, the general trend will be for the size of the remaining portion of the program to decrease as more and more of the program is debugged. (*emphasis added*)

As the above text makes clear, as the debugging of the program proceeds, in general, the first portion of the program, i.e., that portion converted to a hardware configuration program and deployed to the programmable hardware element, grow larger as the process iterates, i.e., for at least a subset of the iterations. It is this aspect that is denoted by the cited phrase in the independent claims. As the text further explains, in some cases, some of the previously debugged parts of the program (that have been previously included in the first portion of the program) may require further debugging, and thus may be moved back to the remaining portion for additional debugging on the computer system. In these cases, the first portion of the program, i.e., that portion converted to a hardware configuration program and deployed to the programmable hardware element, grows smaller over one or more iterations. It is this aspect of the claimed invention that is denoted by the phrase in dependent claims 3, 26, 30, 34, and 38.

The Office Action asserts that such functionality “would never be construed as increasing in size in a automated and programmatic fashion via successive increment (*emphasis added*) as claimed [sic]”, that “the limitation thus cannot be construed semantically as a concrete and repeatable step (i.e., whereby incremental occurs by succession) [sic]”, and that “In order to provide a proper method claim (according to 35 USC. section 112, first, 35 USC section 101), there should be no action step that depends on the aleatory and/or unpredictable factor played by a human intervention as disclosed (e.g., *the user may, may be temporarily, may increase in size*).”.

Applicant respectfully disagrees.

The independent claims include the limitation “receiving user input modifying the remaining portion of the program to debug the remaining portion of the program”. Applicant notes that, per the Specification, such modification can certainly include adding to, or removing from, the remaining portion of the program, i.e., moving code from the remaining portion to the first portion, or moving code from the first portion to the remaining portion, thereby making the first portion successively larger or smaller,

respectively. Thus, Applicant submits that the claim language is not only clear, but is fully supported by the Specification.

Applicant thus respectfully submits that the Specification clearly supports the language of the independent claims in question, as well as the language of dependent claims 3, 26, 30, 34, and 38.

Applicant further respectfully notes that the Specification describes numerous embodiments in which various features may or may not be included; however, the independent claims as written are clearly directed to embodiments where user input modifying the remaining portion of the program to debug the remaining portion of the program *is* received, and where for one or more iterations the first portion of the program *is* a successively larger portion of the program. Thus, there is no “unpredictable factor played by a human intervention” in the claims.

Applicant thus respectfully requests removal of the objection to claims 1, 23, 28, 32, and 36.

Double Patenting Rejection

Claim 6 was rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claim 85 of U.S. Patent No. 7,024,660 (“’660”), specifically, that claim 85 of ‘660 anticipates the features and limitations of claim 6.

The Examiner states that the *repeating* step of amended claim 1 would overcome the double patenting rejection of claim 6, but for the above objection for improper language use. Applicant has addressed the objection above, and thus requests removal of the double patenting rejection.

Section 102 Rejections

Claims 1, 3-7, 9-10, 12-24, 26-28, 30-32, 34-36, and 39-39 were rejected under 35 U.S.C. 102(b) as being anticipated by Tseng et al. (USPN: 6,009,256, “Tseng”). Applicant respectfully disagrees.

As noted above, anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim. *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). The identical invention must be shown in as complete detail as is contained in the claims. *Richardson v. Suzuki Motor Co.*, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989).

Claim 1 recites:

1. A memory medium comprising program instructions for debugging a program, wherein the program is intended for deployment on a programmable hardware element to perform a function, wherein the program instructions are executable to perform:

a) converting a first portion of the program into a first hardware configuration program which is deployable on the programmable hardware element to perform a corresponding first portion of the function, wherein a remaining portion of the program is to be debugged by a user;

b) configuring the programmable hardware element with the first hardware configuration program;

c) executing the program, wherein said executing comprises:

the programmable hardware element executing the first portion of the program; and

the computer system executing the remaining portion of the program;

wherein the remaining portion of the program is operable to be analyzed and debugged in response to said executing;

d) receiving user input modifying the remaining portion of the program to debug the remaining portion of the program; and

repeating a) – d) one or more times in an iterative manner, wherein for one or more iterations the first portion of the program is a successively larger portion of the program.

Applicant respectfully submits that the Examiner has mischaracterized the cited art of Tseng, and has improperly read into Tseng various novel features and limitations of Applicant's claim 1, absent any evidence that Tseng actually discloses these features. In the previous Response, which is hereby incorporated by reference, Applicant provided arguments explaining numerous distinctions between Tseng and Applicant's invention as recited in claim 1, in response to which the Examiner has provided additional arguments. Applicant presents the following remarks, recapitulating previous arguments, and further addressing the Examiner's new arguments.

Applicant submits that Tseng nowhere teaches or suggests **repeating a) – d) one or more times in an iterative manner, wherein for one or more iterations the first portion of the program is a successively larger portion of the program**, as recited in amended claim 1.

Nowhere does Tseng disclose performing such converting, configuring, executing, and receiving user input modifying the remaining portion of the program, in an iterative fashion, where for at least some of the iterations, the first portion of the program is a successively larger portion of the program. In other words, Tseng fails to teach or suggest such iterative debugging, where, as the program is debugged, the (first) portion of the program that is deployed to the programmable hardware element increases in size, and thus, the remaining portion (executed by the computer system and debugged by the user) decreases. Said another way, in Applicant's invention as represented in claim 1, and as clearly described in detail in the Specification, during the iterative debugging process, as the program is debugged, the debugged portions are moved from software implementation to hardware implementation, i.e., from the remaining portion to the first portion, thereby increasing the size of the first portion. Dependent claim 4 expresses the final result of such iterative incremental migration to the hardware portion (the first portion) from the software portion (the remaining portion), where, after the debugging process is complete, the entire program is converted and deployed to the programmable hardware element.

In asserting that Tseng teaches such iteration, the Examiner cites Figures 2, 5, col. 15:34-57, and col.35:65 – col.36:50. However Applicant has reviewed these citations

(and Tseng in general) carefully, and can find no description of these features. For example, neither Figure 2 nor Figure 5 discloses the claimed iterative debugging where the portion deployed to hardware increases in size, i.e., where portions of the program are successively moved from software emulation to hardware deployment as they are debugged. Rather, these figures (and accompanying text) disclose switching back and forth between a software model and a hardware model to debug the circuit design, where the software model and the hardware model remain as originally defined or partitioned (see Figure 4 and accompanying text).

Neither figure discloses successively changing the size of the first portion, i.e., successively changing the partition between the portion of the program executing on the computer system and the portion of the program implemented in hardware. Rather, as Tseng makes clear, the user partitions the program once (again, see Figure 4 and corresponding text), then switches between software simulation and hardware emulation as desired. This point is clearly made in col.16:55-62:

The reconfigurable hardware boards 250 contain the user's emulated circuit design. This SEmulation system has the ability to let the user selectively switch between software simulation and hardware emulation, as well as stop either the simulation or emulation process at any time, cycle-by-cycle, to inspect values from every component in the model, whether register or combinational.

Cited col. 15:34-57 describes emulating the circuit in the context of its target system environment. While this text does mention running the emulation multiple times to debug the circuit design, the text (and Tseng in general) fails to disclose moving parts of the program from software emulation to hardware deployment as they are debugged, i.e., increasing the size of the portion deployed to the programmable hardware element. Rather, as the text describes, “After the emulation with the target system, the user has results that validate the circuit design or reveal non-functional aspects. At this point, the user can simulate/emulate again as indicated at step 135, stop altogether to modify the circuit design, or proceed to integrated circuit fabrication based on the validated circuit design.”

Col.35:65 – col.36:50 describes logging operations for Tseng’s debugging process, and makes no mention of iterative debugging where the portion of the program

deployed to hardware is made successively larger as the program is debugged, i.e., where the program is incrementally moved from software emulation to hardware deployment during debugging.

Thus, based on the Examiner's asserted equivalence of Tseng's hardware and software models as the first portion and remaining portion of claim 1, nowhere does the cited text or Figures (or Tseng in general) disclose iteratively modifying the partition between the software model and the hardware model, where for one or more iterations, the hardware model grows larger (and thus the software model smaller).

The Examiner asserts that Tseng's iterations teach Applicant's claimed repeating, stating that Applicant's claimed limitation that "for one or more iterations the first portion of the program is a successively larger portion of the program" has not been given significant weight because of the Examiner's Objection to the claim language. However, these issues have been resolved above, and so Applicant respectfully submits that this novel limitation of claim 1 should be given its full weight.

Thus, Applicant respectfully submits that Tseng fails to teach or suggest this limitation of claim 1.

Applicant further respectfully submits that Tseng fails to teach or suggest debugging a program, **wherein the program is intended for deployment on a programmable hardware element to perform a function**, as recited in claim 1.

Applicant respectfully notes that Tseng is directed to electronic circuit design testing and simulation/emulation using software and hardware based models. Applicant respectfully notes that the model of the circuit design being tested in Tseng, which the Office Action has interpreted as Applicant's program, is *not* intended for deployment to a programmable hardware element, but rather is solely for the purpose of debugging the design, after which the corresponding circuit will presumably be manufactured. Nowhere does Tseng indicate that the model of the circuit design is intended for deployment on a programmable hardware element (e.g., an FPGA), but rather, the model of the circuit design is implemented in software and/or hardware, for debugging purposes only, and the design is intended for deployment as a circuit, e.g., not for deployment to an FPGA.

In other words, in Tseng's system, both the software and hardware models are for simulating/emulating an electronic circuit design that is *not* intended for deployment on the hardware (FPGA) used to implement the hardware model. Moreover, as noted above, once the user has partitioned the circuit model into a software model and a hardware model, this partition remains in effect. In fact, even after the model has been thoroughly debugged, it is *not* deployed to the FPGA, but rather is sent elsewhere for manufacture of the circuit.

Tseng's system operates in various modes, which are described by Tseng thusly:
Col. 8:51-58:

A. SIMULATION/HARDWARE ACCELERATION MODES

The SEmulator system, through automatic component type analysis, can model the user's custom circuit design in software and hardware. The entire user circuit design is modeled in software, whereas evaluation components (i.e., register component, combinational component) are modeled in hardware. Hardware modeling is facilitated by the component type analysis.

As may be seen, in this mode, the entire user circuit design is modeled in software, and evaluation (testing) components for testing the design are modeled or implemented in hardware (FPGA(s)). Nowhere does the text describe Applicant's claimed portions of a program that is intended for deployment on a programmable hardware element being distributed between a computer system and a programmable hardware element, where the portion on the computer system is to be debugged by a user, nor executing the two portions on the programmable hardware element and computer systems, respectively, as claimed. Applicant notes that Tseng's evaluation components for testing the design (which are implemented in hardware) are *not* part of the program proper, as it is with claim 1, but rather are only used for debugging. More specifically, the software model (which models the entire circuit) is nowhere described as intended for deployment to a programmable hardware element (e.g., an FPGA).

Thus, this mode of Tseng does not anticipate this feature of claim 1.

Col. 9:45-53:

B. EMULATION WITH TARGET SYSTEM MODE

The SEmulation system is capable of emulating the user's circuit within its target system environment. The target system outputs data to the hardware model for evaluation and the hardware model also outputs data to the target system. Additionally, the software kernel controls the operation of this mode so that the user still has the option to start, stop, assert values, inspect values, single step, and switch from one mode to another.

As the text indicates, in this mode, the circuit design is modeled (emulated) within its target system environment. The hardware model (implemented on the FPGA(s)) performs evaluation functionality for testing the circuit, and a software kernel (which controls the operation of the entire system) provides a user interface for the testing system. Clearly, the circuit design is intended for deployment on the target system. Nowhere does the text describe this mode as including a program that is intended for deployment on a programmable hardware element, as claimed. In fact, Tseng clearly describes communication between the target system and the hardware model, which is implemented on the FPGA. Thus, the target system and the FPGA are not, and cannot be, the same. Thus, in this mode, Tseng's models are for debugging a circuit design that is intended for deployment to a target system distinct from the programmable hardware, and thus *not* intended for deployment to programmable hardware; the FPGA(s) is *only* used for debugging.

Thus, this mode also does not anticipate this feature of claim 1.

Col. 9:54-67:

C. POST-SIMULATION ANALYSIS MODE

Logs provide the user with a historical record of the simulation session. Unlike known simulation systems, the SEmulation system does not log every single value, internal state, or value change during the simulation process. The SEmulation system logs only selected values and states based on a logging frequency (i.e., log 1 record every N cycles). During the post-simulation stage, if the user wants to examine various data around point X in the just-completed simulation session, the user goes to one of the logged points, say logged point Y, that is closest and temporally located prior to point X. The user then simulates from that selected logged point Y to his desired point X to obtain simulation results.

As may be seen, this mode is directed to logging selected values and states based on a specified logging frequency. Clearly, this mode similarly fails to anticipate these features and limitations of claim 1.

Col. 10:1-9:

D. HARDWARE IMPLEMENTATION SCHEMES

The SEmulation system implements an array of FPGA chips on a reconfigurable board. Based on the hardware model, the SEmulation system partitions, maps, places, and routes each selected portion of the user's circuit design onto the FPGA chips. Thus, for example, a 4.times.4 array of 16 chips may be modeling a large circuit spread out across these 16 chips.

As the text clearly shows, this aspect of Tseng is directed to distribution of a hardware model over an array of FPGA chips on a reconfigurable board. Again, nowhere does the text disclose Applicant's claimed portions of a program that is intended for deployment on a programmable hardware element being distributed between a computer system and a programmable hardware element, where the portion on the computer system is to be debugged by a user, nor executing the two portions on the programmable hardware element and computer system, respectively, as claimed. Rather, in this mode of Tseng, the entire circuit model is implemented in FPGAs.

Cited Figures 22-33 are directed to implementing the hardware model of a circuit design on an array of FPGAs for debugging purposes, and nowhere disclose debugging a portion of a program on a computer system, where *the program is intended for deployment on a programmable hardware element to perform a function*, as claimed. In other words, in Tseng, the circuit design may be designed to perform a function, but not as an implementation on an FPGA; rather, a *model* of the circuit is implemented on the FPGA for debugging purposes only. In direct contrast, in Applicant's system as represented in claim 1, the program is intended for deployment to a programmable hardware element, and a portion is implemented on the programmable hardware element, while another portion (also intended for deployment on the programmable hardware element) remains on the computer system for debugging purposes, but moved to hardware as it is debugged.

Thus, this aspect of Tseng also does not anticipate these features and limitations of claim 1.

Thus, in each of Tseng's described modes of operation, the user partitions the model into a software model and a hardware model, deploying the hardware model onto the FPGA (for hardware acceleration), and executing the model. Only the hardware model is executed on the FPGA. Nowhere does Tseng disclose moving parts of the program from the software portion to the hardware portion during the debugging process; rather, once the software model and the hardware model have been defined, they remain as originally defined or partitioned. In other words, the portion modeled in software, which in some cases may be the entire circuit, and the portion of the circuit modeled in hardware, e.g., for hardware acceleration, remains the same throughout the debugging process. Thus, while in Applicant's system, the entire program, originally partitioned between hardware and software (the first portion and the remaining portion), is eventually deployed entirely to the programmable hardware element, whereas in Tseng's system, the original partition between software and hardware remains constant.

Thus, in cases where Tseng's model is partitioned between software and hardware, it remains so, and so is not "intended for deployment on a programmable hardware element to perform a function". As discussed above, in cases where the circuit is only modeled in hardware (e.g., mode D), there is no "remaining portion". Thus, either way, Tseng fails to teach all the features and limitations of claim 1.

Thus, none of the disclosed modes of Tseng's system and method anticipate the features and limitations of claim 1. Even in combination, these disclosed modes of operation fail to teach Applicant's invention as represented in claim 1, at least for the reason that Tseng nowhere teaches iteratively increasing the portion of the program deployed to a programmable hardware element during debugging.

The Examiner notes that Tseng allows the user to change register values for a rerun of the simulation/emulation, and asserts that this modifies the "first portion", and that this "reads on remaining portion relative to the first portion". Applicant does not understand what the Examiner means. Tseng's "register components are simple storage components", as Tseng states in col.17:42, and are typically part of the hardware model. Applicant respectfully submits that changing a value in a register is not equivalent to

Applicant's claimed modifying the remaining portion of the program to debug the remaining portion of the program.

Thus, for at least the reasons provided above, Applicant respectfully submits that Tseng neither teaches nor suggests all the features and limitations of claim 1, and so claim 1 and those claims dependent therefrom are patentably distinct and non-obvious over the cited art, and are thus allowable.

Independent claims 23, 28, 32, and 36 include similar limitations as claim 1, and so the above arguments apply with equal force to these claims. Thus, for at least the reasons provided above, claims 23, 28, 32, and 36, and those claims respectively dependent therefrom, are patentably distinct and non-obvious over the cited art, and are thus allowable.

Independent claim 27 also recites limitations similar to those of claim 1, although in a slightly different form. More specifically, claim 27 explicitly recites two iterations of the iterative debugging process, where, as in claim 1, the program is partitioned into two portions, a first portion directed to a programmable hardware element, and a first remaining portion directed to a computer system for debugging by the user. The program is executed, including the programmable hardware element executing the first portion of the program, and the computer system executing the first remaining portion of the program. The remaining portion of the program is analyzed and debugged, e.g., by the user. After the user has debugged (modified) the remaining portion (in response to executing the portions on the programmable hardware element and computer system, respectively), the user repartitions the program, specifying a second portion for deployment on the programmable hardware element that includes the first portion and a debugged portion of the first remaining portion of the program, and a second remaining portion that includes only a subset of the first remaining portion of the program. The second portion is then converted and deployed to the programmable hardware element, and the program executed, where the second portion is executed on the programmable hardware element, and the second remaining portion is executed by the computer system.

Nowhere does Tseng teach or suggest these features and limitations.

The Examiner again relies on the Objection to Applicant's claim language "for one or more iterations the first portion of the program is a successively larger portion of

the program” in the rejection of claim 27. However, claim 27 does not use this language. Rather, as explained above, claim 27 explicitly describes the movement of a debugged program portion from the remaining portion of the program to the first portion of the program (and thus, from software simulation/emulation to hardware implementation) during the debugging process. Thus, the Objection, directed to claims 1, 23, 28, 32, and 36, addressed above, is not germane to claim 27.

As argued at length above, Tseng nowhere discloses this feature.

The Examiner provides no arguments against the patentability of claim 27 directly, but refers instead to arguments presented regarding claims 2 and 3. However, claim 2 has been cancelled, and claim 3 recites the limitation “wherein for one or more other iterations the first portion of the program is a successively smaller portion of the program”, which is not at all equivalent to the subject matter of claim 27, since claim 3 is directed to iterations in which a part of the program is moved *back* to the remaining portion from the first portion, e.g., for further debugging. Thus, the Examiner’s rejection of claim 27 is unfounded, incorrect, and improper.

Thus, for at least the reasons provided above, Applicant respectfully submits that Tseng neither teaches nor suggests all the features and limitations of claim 27, and so claim 27 and those claims dependent therefrom are patentably distinct and non-obvious over the cited art, and are thus allowable.

Removal of the section 102 rejection of claim 1, 3-7, 9-10, 12-24, 26-28, 30-32, 34-36, and 38-39 is earnestly requested.

Applicant also asserts that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

In light of the foregoing amendments and remarks, Applicant submits the application is now in condition for allowance, and an early notice to that effect is requested.

If any extensions of time (under 37 C.F.R. § 1.136) are necessary to prevent the above-referenced application(s) from becoming abandoned, Applicant(s) hereby petition for such extensions. The Commissioner is hereby authorized to charge any fees which may be required or credit any overpayment to Meyertons, Hood, Kivlin, Kowert & Goetzel P.C., Deposit Account No. 50-1505/5150-79600/JCH.

Also filed herewith are the following items:

- ☐ Request for Continued Examination
- ☐ Terminal Disclaimer
- ☐ Power of Attorney By Assignee and Revocation of Previous Powers
- ☐ Notice of Change of Address
- ☐ Other:

Respectfully submitted,

/Jeffrey C. Hood/

Jeffrey C. Hood, Reg. #35198
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel PC
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800
Date: 2007-04-26 JCH/MSW